THEOREM∀
and
Mathematical Knowledge Management

Bruno Buchberger
RISC
(Research Institute for Symbolic Computation)
Johannes Kepler University, Linz, Austria
Bruno.Buchberger@RISC.Uni-Linz.ac.at

Talk at the IMA Workshop "Special Functions in the Digital Age"
August 1, 2002, University of Minnesota,Minneapolis, USA

The Simple Message

Summary

- Function Tables are Mathematical Knowledge.
- Function Tables are "Digitized".
- There is Lots of Mathematical Knowledge Other Than Function Tables.

All Mathematical Knowledge should be "Digitized".

"Function Tables" (Formulae Handbooks)

Example:

$$\sum_{k=1,\ldots,n}\left(\frac{-(-1)^k\,2^{-2k}\,(2k)!\,(1+4k)}{k!\,(1+k)!\,((-1)+2k)}\right)=1+\frac{-(-1)^n\,2^{-2n}\,(2n)!}{n!\,(1+n)!}$$

See: P.Paule, Computer-Solution of Problem 94-2, SIAM REVIEW Vol.37 (1995),105-106.

This is a "formula": equality, inequality, ... with a couple of free variables.

What Does "Digitized" Mean for Function Tables ?

Basic Meaning:

Formulae accessible (over the web) in some machine-readable text format (TeX, MathML, etc.).

Hard enough!

Extended Meaning I:

Formulae can be used (evaluated, applied for simplification of other formulae, etc.) applying certain (simple, ...) algorithms.

Example: Integral tables as part of integration "algorithm" in math systems like Macsyma, Maple, Mathematica, etc.

Extended Meaning II:

Formulae (from a certain infinite class) can be invented and / or proved by an algorithm.

Such algorithms (like Risch' algorithm for integration, or Gosper-Zeilberger-Paule-... method for summation) are based on mathematical theorems that go beyond the content of the formulae to be invented / verified.

Such algorithms are based on "meta-theorems".

Over the past decades, tremendous progress has been made in the mathematics on which the algorithmic invention / proof of formulae in "funtion tables" is based.

General Mathematical Knowledge

The Class of Predicate Logic Formulae

Mathematical knowledge is much more than the formulae contained in math tables and handbooks.

As a matter of fact, all mathematical knowledge can be (conveniently) formulated in predicate logic:

$$\text{limit}[f, a] \Longleftrightarrow \forall_{\epsilon > 0} \exists_{N} \forall_{n \geq N} |f[n] - a| < \epsilon$$

$$\text{is–topology}[P, \mathcal{U}] \Longleftrightarrow \bigwedge \begin{cases} \forall_{p \in P} \forall_{N \in \mathcal{U}[p]} N \in P \\ \forall_{p \in P} \forall_{N \in \mathcal{U}[p]} p \in N \\ \forall_{p \in P} \forall_{M \in \mathcal{U}[p], N \subseteq P} (M \subseteq N \Rightarrow N \in \mathcal{U}[p]) \\ \forall_{p \in P} \forall_{M, N \in \mathcal{U}[p]} M \cap N \in \mathcal{U}[p] \\ \forall_{p \in P} P \in \mathcal{U}[p] \\ \forall_{p \in P} \forall_{M \in \mathcal{U}[p]} \exists_{N \in \mathcal{U}[p]} \forall_{n \in N} M \in \mathcal{U}[n] \end{cases}$$

$$\forall_{F \subseteq \mathcal{P}} \exists_{G \subseteq \mathcal{P}} ((\text{Ideal}[F] = \text{Ideal}[G]) \wedge \text{is–Groebner–basis}[G])$$

In general, formulae expressing mathematical knowledge may contain quantifiers (∀, ∃, ...).

Typically, formulae in math tables and handbooks are quantifier-free. (What about the Σ-quantifier ?)

■ The Fundamental Interplay Between Quantifier Math and Quantifier-Free Math

The part of mathematics that can be expressed by quantifier-free predicate logic is the "symbolic computation" part of mathematics. ("High school" proving, "manipulation" of formulae by "transformation", substitution and replacement, ...)

It contains "programming and computation" as a special case.

Mathematics in "non–algorithmic", "abstract" domains typically starts with formulae that contain quantifiers.

Example: definition of limit:

$$\text{limit}[f, a] \Longleftrightarrow \forall_{\epsilon > 0} \exists_{N} \forall_{n \geq N} |f[n] - a| < \epsilon$$

The first steps in the exploration of a mathematical notion (whose definition involves quantifiers) aim at proving theorems on the interaction of the new notion with known notions that can be expressed quantifier-free.

Example:

$$\text{limit}[f, a] \wedge \text{limit}[g, b] \Longrightarrow \text{limit}[f + g, a + b]$$

The proof of these theorems needs full quantifier-logic.

In contrast, the application of these theorems, only uses quantifier-free logic ("symbolic computation").

In other words, the goal of mathematical exploration is

- to prove by quantifier-logic
- quantifier-free formulae
- about notions defined in quantifier-logic
- in order that, in the future, the application of these theorems can proceed quantifier-free.

What Does "Digitized" Mean for General Mathematical Knowledge ?

Basic Meaning:

Formulae accessible (over the web) in some machine-readable text format (tex, MathML, etc.).

See, for example, the MathWorld project by Eric Weisstein accessible via the Mathematica web site.

(Note: "Basic" does not mean "useless" nor "effortless".)

Extended Meaning I:

Formulae can be used (retrieved, compared, evaluated, etc.) applying algorithms.

Example:

> Given: A definition of "topology" in a slightly different formulation and notation.
>
> Question: Is this definition "contained" in a given knowledge base, i.e. is this definition equivalent with the one in the knowledge base?

In the context of general math knowledge, these goals are not easy to achieve:

Essentially, the algorithmic solution of these questions needs automated theorem proving (both in quantifier-free and quantifier-logic).

Extended Meaning II:

Formulae (from a certain infinite class of quantifier logic) can be invented and / or proved by an algorithm.

There are theorem proving algorithms

- for the class of all predicate logic formulae (e.g. Robinson's resolution method)
- for various special classes (formulae in special "theories"), e.g. the theory of reald closed fields (Collins' algorithm).

As in the case of function tables, such algorithms are based on "meta-theorems".

(The formulae in function tables are a special class of math formulae).

Mathematics can be viewed as a network of meta-theories: A theorem on a meta-level may "trivialize" the invention / proof of infinitely many theorems on the object level.

Tremendous progress in "trivializing" the invention / proof of "formulae" has been made in the past few decades.

(The algorithms for inventing / proving certain classes of formulae in function tables are one example of such trivializing algorithms. Others are Collins' algorithm for the theory of real closed fields or the Groebner bases method for geo proving.)

The trivialization of the invention / proof of a particular (infinite) class of formulae, typically, is a highly nontrivial task.

Ingredients of Future "Digitized" Mathematical Knowledge Management Systems

One Logic Frame

Formulae must be expressions within a formal language / logic and not only "texts".

Preferrably, there should be one uniform logic frame for all of mathematics.

Predicate logic is a natural candidate for such a universal logic frame.

(If we allow various logics, mutual translation / interpretation must be provided.)

■ Math Knowledge Bases

All mathematical knowledge must be formalized within this language / logic frame.

By Bourbakism, one knows that (even first order) predicate logic is a suitable language for all of mathematics.

In fact, predicate logic is also a convenient and elegant language.

(Side remark: Strangely enough, practically, the Bourbaki volumes are *not* good examples of well formalized texts!)

(Side remark: The formal abilities and the formal culture of (some,many, ...?) current mathematicians are not mature enough for this goal.)

■ Proving, Solving, Simplifying (Computing)

These are the key activities of mathematics.

They must be expressible in one logic / system frame.

Predicate logic is a natural frame for these three aspects.

Current systems do not integrate the three aspects.

■ Structured Knowledge Bases

The key to success in mathematical exploration is to proceed in exploration layers.

One exploration layer:

- Given notions and knowledge about these notions.
- A new notion is introduced by describing its relation with the given notions (definitions, axioms).
- Exploration of new knowledge: All the possible interactions of the new notion with the given notions and with itself are explored by proving and added to the knowledge base.
- Termination criterion: new knowledge is "complete" when all other consequences can be proved by quantifier-free logic.

Example:

$$(\text{gcd}[x, y] \mid x) \bigwedge (\text{gcd}[x, y] \mid y) \bigwedge \forall_z ((z \mid x \wedge z \mid y) \Longrightarrow z \mid \text{gcd}[x, y])$$

Such axioms define a "functor" that describes a new domain (with operation "gcd") in dependence on a given domain (with operation "|").

A functor does not only map operations into operations but also properties of operations in to properties of operations.

■ Automated Proving, Solving, Simplifying

The core of future Mathematical Knowledge Management Systems:

- knowledge bases for various theories
- libraries of algorithmic provers, solvers, simplifiers for many theories
- facilities for extending existing provers, ... and for programming new provers, ...

■ Formula Retrieving = A Special Case of Proving

A formula is retrievable from a knowledge base if it is "easily" provable from the knowledge base.

■ Current "Digitized" Mathematical Knowledge Management Systems

There are basically no systems that have all the above ingredients:

- function tables etc: confined to quantifier-free knowledge
- "math software systems" like Mathematica, Maple, ...: no proving
- proof checkers like Mizar: no automated proof generation (Mizar is the only system with a huge knowledge base)
- automated theorem proving systems like Otter: no solving, programming and computing, no knowledge bases.

September 2001: 1st International Conference on MKM at RISC. European Consortium for MKM formed.

July 2002: North American Conference on MKM at McMaster U, Canada. US Consortium for MKM formed.

February 2003: 2nd International Conference on MKM in Bologna.

Theorema

An attempt at a math knowledge management system with all the above ingredients:

- full predicate logic as the language / logic frame,
- programmed in *Mathematica,* i.e. platform-independent; however, does not use implicit knowledge of Mathematica,
- heavily uses the Mathematica front-end,
- implements a growing library of new and known methods for computer-supported proving, solving, simplifying for various areas of mathematics,
- *Theorema* Group: Bruno Buchberger (project leader).
  Senior researchers: Tudor Jebelean, Wolfgang Windsteiger, Temur Kutsia, and Koji Nakagawa,
  Current PhD students: Markus Rosenkranz, Florina Piroi, Adrian Craciun.
  Former PhD students: Elena Tomuta, Daniela Vasaru, Claudio Dupret.

Example: Manipulating Mathematical Text

Definition["limit:", any[f, a],

$$\text{limit}[f, a] \Longleftrightarrow \underset{\epsilon>0}{\forall_{\epsilon}} \underset{N}{\exists} \underset{n\geq N}{\forall_{n}} |f[n]-a|<\epsilon]$$

```
Definition["limit:"] // InputForm
```

```
•def["limit:", •range[
  •simpleRange[•var[f]],
  •simpleRange[•var[a]]],
 True, •flist[•lf["",
  ™Iff[limit[•var[f],
    •var[a]], ™ForAll[
    •range[•simpleRange[
      •var[∈]]],
    ™Greater[•var[∈],
     0], ™Exists[•range[
      •simpleRange[
       •var[™N]]], True,
     ™ForAll[•range[
       •simpleRange[
        •var[n]]],
      ™GreaterEqual[
       •var[n], •var[
        ™N]], ™Less[
       ™BracketingBar[
        ™Minus[•var[f][
         •var[n]],
         •var[a]]],
       •var[∈]]]]]]]]]
```

Proposition["limit of sum", any[f, a, g, b],
(limit[f, a] ∧ limit[g, b]) ⇒ limit[f + g, a + b]]

Definition["+:", any[f, g, x],
(f + g)[x] = f[x] + g[x]]

Lemma["|+|", any[x, y, a, b, δ, ϵ],
(|(x + y) − (a + b)| < (δ + ϵ)) ⟸ (|x − a| < δ ∧ |y − b| < ϵ)]

Lemma["max", any[m, M1, M2],
m ≥ max[M1, M2] ⇒ (m ≥ M1 ∧ m ≥ M2)]

Theory["limit",
Definition["limit:"]
Definition["+:"]
Lemma["|+|"]
Lemma["max"]
]

Example: Some Provers

Example: The PCS Prover

A heuristic proof method (BB 2000) for predicate logic, in particular, for formulae with "alternativing quantifiers".

Generates "natural" proofs.

Reduces proving to solving.

```
Prove[Proposition["limit of sum"], using → Theory["limit"], by → PCS]
```

```
- ProofObject -
```

Proof contains interesting algorithmic and didactic information!

Such proofs seem to be trivial but they are a challenge for automated proving.

Proof generated completely automatically by the above prover algorithm (PCS) call:

Prove:

(Proposition (limit of sum)) $\forall_{f,a,g,b} (\text{limit}[f, a] \wedge \text{limit}[g, b] \Rightarrow \text{limit}[f + g, a + b])$,

under the assumptions:

(Definition (limit:)) $\forall_{f,a} \left(\text{limit}[f, a] \Leftrightarrow \forall_{\epsilon, \epsilon>0} \exists_{N} \forall_{n, n \geq N} (|f[n] - a| < \epsilon) \right)$,

(Definition (+:)) $\forall_{f,g,x} ((f + g)[x] = f[x] + g[x])$,

(Lemma (|+|)) $\forall_{x,y,a,b,\delta,\epsilon} (|(x + y) - (a + b)| < \delta + \epsilon \Leftarrow (|x - a| < \delta \wedge |y - b| < \epsilon))$,

(Lemma (max)) $\forall_{m,M1,M2} (m \geq \max[M1, M2] \Rightarrow m \geq M1 \wedge m \geq M2)$.

We assume

(1) $\text{limit}[f_0, a_0] \wedge \text{limit}[g_0, b_0]$,

and show

(2) $\text{limit}[f_0 + g_0, a_0 + b_0]$.

Formula (1.1), by (Definition (limit:)), implies:

(3) $\underset{\epsilon>0}{\forall}\ \underset{N}{\exists}\ \underset{n\geq N}{\forall}\ (|f_0[n] - a_0| < \epsilon)$.

By (3), we can take an appropriate Skolem function such that

(4) $\underset{\epsilon>0}{\forall}\ \underset{n\geq N_0[\epsilon]}{\forall}\ (|f_0[n] - a_0| < \epsilon)$,

Formula (1.2), by (Definition (limit:)), implies:

(5) $\underset{\epsilon>0}{\forall}\ \underset{N}{\exists}\ \underset{n\geq N}{\forall}\ (|g_0[n] - b_0| < \epsilon)$.

By (5), we can take an appropriate Skolem function such that

(6) $\underset{\epsilon>0}{\forall}\ \underset{n\geq N_1[\epsilon]}{\forall}\ (|g_0[n] - b_0| < \epsilon)$,

Formula (2), using (Definition (limit:)), is implied by:

(7) $\underset{\epsilon>0}{\forall}\ \underset{N}{\exists}\ \underset{n\geq N}{\forall}\ (|(f_0 + g_0)[n] - (a_0 + b_0)| < \epsilon)$.

We assume

(8) $\epsilon_0 > 0$,

and show

(9) $\underset{N}{\exists}\ \underset{n\geq N}{\forall}\ (|(f_0 + g_0)[n] - (a_0 + b_0)| < \epsilon_0)$.

We have to find N_2^* such that

(10) $\underset{n}{\forall}\ (n \geq N_2^* \Rightarrow |(f_0 + g_0)[n] - (a_0 + b_0)| < \epsilon_0)$.

Formula (10), using (Definition (+:)), is implied by:

(11) $\underset{n}{\forall}\ (n \geq N_2^* \Rightarrow |(f_0[n] + g_0[n]) - (a_0 + b_0)| < \epsilon_0)$.

Formula (11), using (Lemma (|+|)), is implied by:

(12) $\exists_{\substack{\delta,\epsilon \\ \delta+\epsilon=\epsilon_0}} \forall_n (n \geq N_2^* \Rightarrow |f_0[n] - a_0| < \delta \wedge |g_0[n] - b_0| < \epsilon)$.

We have to find δ_0^*, ϵ_1^*, and N_2^* such that

(13) $(\delta_0^* + \epsilon_1^* = \epsilon_0) \bigwedge \forall_n (n \geq N_2^* \Rightarrow |f_0[n] - a_0| < \delta_0^* \wedge |g_0[n] - b_0| < \epsilon_1^*)$.

Formula (13), using (6), is implied by:

$(\delta_0^* + \epsilon_1^* = \epsilon_0) \bigwedge \forall_n (n \geq N_2^* \Rightarrow \epsilon_1^* > 0 \wedge n \geq N_1[\epsilon_1^*] \wedge |f_0[n] - a_0| < \delta_0^*)$,

which, using (4), is implied by:

$(\delta_0^* + \epsilon_1^* = \epsilon_0) \bigwedge \forall_n (n \geq N_2^* \Rightarrow \delta_0^* > 0 \wedge \epsilon_1^* > 0 \wedge n \geq N_0[\delta_0^*] \wedge n \geq N_1[\epsilon_1^*])$,

which, using (Lemma (max)), is implied by:

(14) $(\delta_0^* + \epsilon_1^* = \epsilon_0) \bigwedge \forall_n (n \geq N_2^* \Rightarrow \delta_0^* > 0 \wedge \epsilon_1^* > 0 \wedge n \geq \max[N_0[\delta_0^*], N_1[\epsilon_1^*]])$.

Formula (14) is implied by

(15) $(\delta_0^* + \epsilon_1^* = \epsilon_0) \bigwedge \delta_0^* > 0 \bigwedge \epsilon_1^* > 0 \bigwedge \forall_n (n \geq N_2^* \Rightarrow n \geq \max[N_0[\delta_0^*], N_1[\epsilon_1^*]])$.

Partially solving it, formula (15) is implied by

(16) $(\delta_0^* + \epsilon_1^* = \epsilon_0) \wedge \delta_0^* > 0 \wedge \epsilon_1^* > 0 \wedge (N_2^* = \max[N_0[\delta_0^*], N_1[\epsilon_1^*]])$.

Now,

$(\delta_0^* + \epsilon_1^* = \epsilon_0) \wedge \delta_0^* > 0 \wedge \epsilon_1^* > 0$

can be solved for δ_0^* and ϵ_1^* by a call to Collins cad–method yielding a sample solution

$\delta_0^* \leftarrow \frac{\epsilon_0}{2}$,

$\epsilon_1^* \leftarrow \frac{\epsilon_0}{2}$.

Furthermore, we can immediately solve

$N_2^* = \max[N_0[\delta_0^*], N_1[\epsilon_1^*]]$

for N_2^* by taking

$N_2^* \leftarrow \max[N_0[\frac{\epsilon_0}{2}], N_1[\frac{\epsilon_0}{2}]]$.

Hence formula (16) is solved, and we are done.

□

■ Example: Set Theory Prover

The *Theorema* set theory prover (PhD thesis of W. Windsteiger) is also based on the PCS principle.

Definition["reflexivity", any[~ , A],
  $\text{is–reflexive}_A[\sim] :\Leftrightarrow \forall_{x\in A}\ x \sim x$]

Definition["symmetry", any[~ , A],
  $\text{is–symmetric}_A[\sim] :\Leftrightarrow \forall_{x,y\in A}\ (x \sim y \Rightarrow y \sim x)$]

Definition["transitivity", any[~ , A],
  $\text{is–transitive}_A[\sim] :\Leftrightarrow \forall_{x,y,z\in A}\ (x \sim y \wedge y \sim z \Rightarrow x \sim z)$]

Definition["equivalence", any[~ , A],
  $\text{is–equivalence}_A[\sim] :\Leftrightarrow \bigwedge \begin{Bmatrix} \text{is–reflexive}_A[\sim] \\ \text{is–symmetric}_A[\sim] \\ \text{is–transitive}_A[\sim] \end{Bmatrix}$]

Definition["class", any[x, ~ , A],
  $\text{class}_{A,\sim}[x] := \{a \in A \mid a \sim x \wedge x \in A\}$]

Proposition["equal classes", any[x ∈ A, y ∈ A, ~ , A], with[$\text{is–equivalence}_A[\sim]$],
  $x \sim y \Rightarrow (\text{class}_{A,\sim}[x] = \text{class}_{A,\sim}[y])$]

```
Prove[Proposition["equal classes"],
  using → ⟨Definition["equivalence"], Definition["transitivity"], Definition["symmetry"], Definition["class"]⟩,
  by → SetTheoryPCSProver,
  transformBy → ProofSimplifier, TransformerOptions → {branches → Proved, steps → Useful}, ShowOptions → {},
  ProverOptions → {GRWTarget → {"goal", "kb"}, AllowIntroduceQuantifiers → True,
      UseCyclicRules → True, DisableProver → {STC, PND}, ApplyBuiltIns → {}}, SearchDepth → 50]
```

Proof generated completely automatically by the above set theory prover call:

Prove:

(Proposition (equal classes))

$$\forall_{A,x,y,\sim} (x \in A \land y \in A \land \text{is–equivalence}_A[\ \sim\ ] \Rightarrow (x \sim y \Rightarrow (\text{class}_{A,\sim}[x] = \text{class}_{A,\sim}[y])))$$

under the assumptions:

(Definition (equivalence))

$$\forall_{A,\sim} (\text{is–equivalence}_A[\ \sim\ ] :\Leftrightarrow \text{is–reflexive}_A[\ \sim\ ] \land \text{is–symmetric}_A[\ \sim\ ] \land \text{is–transitive}_A[\ \sim\ ])$$

(Definition (transitivity)) $$\forall_{A,\sim} \left(\text{is–transitive}_A[\ \sim\ ] :\Leftrightarrow \forall_{x,y,z} (x \in A \land y \in A \land z \in A \Rightarrow (x \sim y \land y \sim z \Rightarrow x \sim z))\right)$$

(Definition (symmetry)) $$\forall_{A,\sim} \left(\text{is–symmetric}_A[\ \sim\ ] :\Leftrightarrow \forall_{x,y} (x \in A \land y \in A \Rightarrow (x \sim y \Rightarrow y \sim x))\right)$$

(Definition (class)) $$\forall_{A,x,\sim} \left(\text{class}_{A,\sim}[x] := \left\{a \mid_a (a \sim x \land x \in A) \land a \in A\right\}\right).$$

We assume

(1) $\text{x}_0 \in \text{A}_0 \land \text{y}_0 \in \text{A}_0 \land \text{is–equivalence}_{\text{A}_0}[\ \sim_0\ ]$,

and show

(2) $\text{x}_0 \sim_0 \text{y}_0 \Rightarrow (\text{class}_{\text{A}_0,\sim_0}[\text{x}_0] = \text{class}_{\text{A}_0,\sim_0}[\text{y}_0])$.

We prove (2) by the deduction rule.

We assume

(5) $\text{x}_0 \sim_0 \text{y}_0$

and show

(6) $\text{class}_{\text{A}_0,\sim_0}[\text{x}_0] = \text{class}_{\text{A}_0,\sim_0}[\text{y}_0]$.

Formula (6), using (Definition (class)), is implied by:

(7) $\left\{a \mid_a \text{x}_0 \in \text{A}_0 \land a \in \text{A}_0 \land a \sim_0 \text{x}_0\right\} = \left\{a \mid_a \text{y}_0 \in \text{A}_0 \land a \in \text{A}_0 \land a \sim_0 \text{y}_0\right\}$.

We show (7) by mutual inclusion:

$\subseteq$: We assume

(8) $\text{x}_0 \in \text{A}_0 \land \text{a1}_0 \in \text{A}_0 \land \text{a1}_0 \sim_0 \text{x}_0$

and show:

(9) $y_0 \in A_0 \wedge a1_0 \in A_0 \wedge a1_0 \sim_0 y_0$.

We prove the individual conjunctive parts of (9):

Proof of (9.1) $y_0 \in A_0$:

Formula (9.1) is true because it is identical to (1.2).

Proof of (9.2) $a1_0 \in A_0$:

Formula (9.2) is true because it is identical to (8.2).

Proof of (9.3) $a1_0 \sim_0 y_0$:

Formula (1.3), by (Definition (equivalence)), implies:

$\text{is–reflexive}_{A_0}[\sim_0] \bigwedge \text{is–symmetric}_{A_0}[\sim_0] \bigwedge \text{is–transitive}_{A_0}[\sim_0]$,

which, by (Definition (transitivity)), implies:

(12) $\forall_{x,y,z} (x \in A_0 \wedge y \in A_0 \wedge z \in A_0 \Rightarrow (x \sim_0 y \wedge y \sim_0 z \Rightarrow x \sim_0 z)) \bigwedge \text{is–reflexive}_{A_0}[\sim_0] \bigwedge \text{is–symmetric}_{A_0}[\sim_0]$.

Formula (9.3), using (12.1), is implied by:

(13) $\exists_y (a1_0 \in A_0 \wedge y_0 \in A_0 \wedge y \in A_0 \wedge a1_0 \sim_0 y \wedge y \sim_0 y_0)$.

Now, let $y := x_0$. Thus, for proving (13) it is sufficient to prove:

(14) $a1_0 \in A_0 \wedge y_0 \in A_0 \wedge x_0 \in A_0 \wedge a1_0 \sim_0 x_0 \wedge x_0 \sim_0 y_0$.

We prove the individual conjunctive parts of (14):

Proof of (14.1) $a1_0 \in A_0$:

Formula (14.1) is true because it is identical to (8.2).

Proof of (14.2) $y_0 \in A_0$:

Formula (14.2) is true because it is identical to (1.2).

Proof of (14.3) $x_0 \in A_0$:

Formula (14.3) is true because it is identical to (8.1).

Proof of (14.4) $a1_0 \sim_0 x_0$:

Formula (14.4) is true because it is identical to (8.3).

Proof of (14.5) $x_0 \sim_0 y_0$:

Formula (14.5) is true because it is identical to (5).

$\supseteq$: Now we assume

(9) $y_0 \in A_0 \wedge a1_0 \in A_0 \wedge a1_0 \sim_0 y_0$

and show:

(8) $x_0 \in A_0 \wedge a1_0 \in A_0 \wedge a1_0 \sim_0 x_0$.

We prove the individual conjunctive parts of (8):

Proof of (8.1) $x_0 \in A_0$:

Formula (8.1) is true because it is identical to (1.1).

Proof of (8.2) $a1_0 \in A_0$:

Formula (8.2) is true because it is identical to (9.2).

Proof of (8.3) $a1_0 \sim_0 x_0$:

Formula (1.3), by (Definition (equivalence)), implies:

$\text{is–reflexive}_{A_0}[\sim_0] \bigwedge \text{is–symmetric}_{A_0}[\sim_0] \bigwedge \text{is–transitive}_{A_0}[\sim_0]$,

which, by (Definition (transitivity)), implies:

(19)

$$\forall_{x,y,z} (x \in A_0 \wedge y \in A_0 \wedge z \in A_0 \Rightarrow (x \sim_0 y \wedge y \sim_0 z \Rightarrow x \sim_0 z)) \bigwedge \text{is–reflexive}_{A_0}[\sim_0] \bigwedge \text{is–symmetric}_{A_0}[\sim_0].$$

Formula (8.3), using (19.1), is implied by:

(20) $\exists_y (a1_0 \in A_0 \wedge x_0 \in A_0 \wedge y \in A_0 \wedge a1_0 \sim_0 y \wedge y \sim_0 x_0)$.

Now, let $y := y_0$. Thus, for proving (20) it is sufficient to prove:

(21) $a1_0 \in A_0 \wedge x_0 \in A_0 \wedge y_0 \in A_0 \wedge a1_0 \sim_0 y_0 \wedge y_0 \sim_0 x_0$.

We prove the individual conjunctive parts of (21):

Proof of (21.1) $a1_0 \in A_0$:

Formula (21.1) is true because it is identical to (9.2).

Proof of (21.2) $x_0 \in A_0$:

Formula (21.2) is true because it is identical to (1.1).

Proof of (21.3) $y_0 \in A_0$:

Formula (21.3) is true because it is identical to (9.1).

Proof of (21.4) $a1_0 \sim_0 y_0$:

Formula (21.4) is true because it is identical to (9.3).

Proof of (21.5) $y_0 \sim_0 x_0$:

Formula (19.3), by (Definition (symmetry)), implies:

(24) $\underset{x,y}{\forall} (x \in A_0 \wedge y \in A_0 \Rightarrow (x \sim_0 y \Rightarrow y \sim_0 x))$.

Formula (21.5), using (24), is implied by:

(25) $x_0 \in A_0 \wedge y_0 \in A_0 \wedge x_0 \sim_0 y_0$.

We prove the individual conjunctive parts of (25):

Proof of (25.1) $x_0 \in A_0$:

Formula (25.1) is true because it is identical to (1.1).

Proof of (25.2) $y_0 \in A_0$:

Formula (25.2) is true because it is identical to (9.1).

Proof of (25.3) $x_0 \sim_0 y_0$:

Formula (25.3) is true because it is identical to (5).

□

■ Exampe: Induction Proofs and the Cascade

```
Definition["addition", any[m, n],
        m + 0 = m              " +0:"
        m + n⁺ = (m + n)⁺      " + .:"]
```

```
Proposition["commutativity of addition", any[m, n],
        m + n = n + m   " + = "]
```

Failing proof:

```
Prove[Proposition["commutativity of addition"],
        using → ⟨Definition["addition"]⟩,
        by → NNEqIndProver,
        ProverOptions → {TermOrder → LeftToRight},
    transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved, Failed}}];
```

The above prover call generates a failing inductive proof.

```
Prove[Proposition["commutativity of addition"],
        using → Definition["addition"],
    by → Cascade[NNEqIndProver, ConjectureGenerator], ProverOptions → {TermOrder → LeftToRight}];
```

The above prover call generates a "cascade" of proofs attempts and proofs. Form the failing proof attempts, a "conjecture generator" generates conjectures for lemmata whose proof leads to a successful inductive proof of the main theorem. Here are the five proof attempts and proofs generated in this example:

Proof attempt of theorem and generation of first conjecture:

Prove:

(Proposition (commutativity of addition): + =) $\underset{m,n}{\forall}\ (m + n = n + m)$,

under the assumptions:

(Definition (addition): +0:) $\underset{m}{\forall}\ (m + 0 = m)$,

(Definition (addition): + .:) $\underset{m,n}{\forall}\ (m + n^{+} = (m + n)^{+})$.

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: failed

The proof of (Proposition (commutativity of addition): + =)fails. (The prover "Simplifier" was unable to transform the proof situation.)

Alternative proof 2: failed

We try to prove (Proposition (commutativity of addition): + =) by induction on *m*.

Induction Base:

(1) $\forall_n (0 + n = n + 0)$.

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: proved

We take in (1) all variables arbitrary but fixed:

(4) $0 + n_1 = n_1 + 0$

and simplify it.

Simplification of the lhs term:

$0 + n_1 \rfloor$

Simplification of the rhs term:

$n_1 + 0 =$ by (Definition (addition): +0:)

$n_1 \rfloor$

Hence, it is sufficient to prove:

(5) $\forall_n (0 + n = n)$.

We prove (5) by induction on *n*.

Induction Base:

(6) $0 + 0 = 0$.

A proof by simplification of (6) works.

Simplification of the lhs term:

$0 + 0 =$ by (Definition (addition): +0:)

$0\rfloor$

Simplification of the rhs term:

$0\rfloor$

Induction Step:

Induction Hypothesis:

(7) $0 + n_2 = n_2$

Induction Conclusion:

(8) $0 + n_2^+ = n_2^+$.

A proof by simplification of (8) works.

Simplification of the lhs term:

$0 + n_2^+ =$ by (Definition (addition): + .:)

$(0 + n_2)^+ =$ by (7)

$n_2^+\rfloor$

Simplification of the rhs term:

$n_2^+\rfloor$

Alternative proof 2: pending

Pending proof of (1).

Induction Step:

Induction Hypothesis:

(2) $\underset{n}{\forall}\ (m_1 + n = n + m_1)$

Induction Conclusion:

(3) $\underset{n}{\forall}\ (m_1^+ + n = n + m_1^+)$.

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: failed

We take in (3) all variables arbitrary but fixed:

(9) $m_1^+ + n_3 = n_3 + m_1^+$

and simplify it.

Simplification of the lhs term:

$m_1^+ + n_3 \rfloor$

Simplification of the rhs term:

$n_3 + m_1^+ =$ by (Definition (addition): + .:)

$(n_3 + m_1)^+ \rfloor$

Hence, it is sufficient to prove:

(10) $\underset{n}{\forall} (m_1^+ + n = (n + m_1)^+)$.

We try to prove (10) by induction on *n*.

Induction Base:

(11) $m_1^+ + 0 = (0 + m_1)^+$.

We simplify (11).

Simplification of the lhs term:

$m_1^+ + 0 =$ by (Definition (addition): +0:)

$m_1^+ \rfloor$

Simplification of the rhs term:

$(0 + m_1)^+ \rfloor$

Hence, it is sufficient to prove:

(14) $m_1^+ = (0 + m_1)^+$.

The proof of (14) fails. (No applicable inference rule was found.)

Induction Step:

Induction Hypothesis:

(12) $m_1^+ + n_4 = (n_4 + m_1)^+$

Induction Conclusion:

(13) $m_1^+ + n_4^+ = (n_4^+ + m_1)^+$.

We simplify (13).

Simplification of the lhs term:

$m_1^+ + n_4^+ =$ by (Definition (addition): + .:)

$(m_1^+ + n_4)^+ =$ by (12)

$((n_4 + m_1)^+)^+$⌋

Simplification of the rhs term:

$(n_4^+ + m_1)^+$⌋

Hence, it is sufficient to prove:

(15) $((n_4 + m_1)^+)^+ = (n_4^+ + m_1)^+$.

The proof of (15) fails. (No applicable inference rule was found.)

Alternative proof 2: failed

We try to prove (3) by induction on *n*.

Induction Base:

(16) $m_1^+ + 0 = 0 + m_1^+$.

We simplify (16).

Simplification of the lhs term:

$m_1^+ + 0 =$ by (Definition (addition): +0:)

m_1^+⌋

Simplification of the rhs term:

$0 + m_1^+ =$ by (Definition (addition): + .:)

$(0 + m_1)^+ \rfloor$

Hence, it is sufficient to prove:

(19) $m_1^+ = (0 + m_1)^+$.

The proof of (19) fails. (No applicable inference rule was found.)

Induction Step:

Induction Hypothesis:

(17) $m_1^+ + n_5 = n_5 + m_1^+$

Induction Conclusion:

(18) $m_1^+ + n_5^+ = n_5^+ + m_1^+$.

Pending proof of (18).

□

Proof of first conjecture:

Prove:

(Proposition (20): 20) $\forall_m (0 + m = m)$,

under the assumptions:

(Definition (addition): +0:) $\forall_m (m + 0 = m)$,

(Definition (addition): + .:) $\forall_{m,n} (m + n^+ = (m + n)^+)$.

We prove (Proposition (20): 20) by induction on *m*.

Induction Base:

(1) $0 + 0 = 0$.

A proof by simplification of (1) works.

Simplification of the lhs term:

$0 + 0 =$ by (Definition (addition): +0:)

$0\rfloor$

Simplification of the rhs term:

$0\rfloor$

Induction Step:

Induction Hypothesis:

(2) $0 + m_1 = m_1$

Induction Conclusion:

(3) $0 + m_1^+ = m_1^+$.

A proof by simplification of (3) works.

Simplification of the lhs term:

$0 + m_1^+ =$ by (Definition (addition): + .:)

$(0 + m_1)^+ =$ by (2)

$m_1^+\rfloor$

Simplification of the rhs term:

$m_1^+\rfloor$

□

Proof attempt and generation of second conjecture:

Prove:

(Proposition (commutativity of addition): + =) $\forall_{m,n} (m + n = n + m)$,

under the assumptions:

(Proposition (20): 20) $\forall_m (0 + m = m)$,

(Definition (addition): +0:) $\forall_m (m + 0 = m)$,

(Definition (addition): + .:) $\forall_{m,n} (m + n^+ = (m + n)^+)$.

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: failed

The proof of (Proposition (commutativity of addition): + =)fails. (The prover "Simplifier" was unable to transform the proof situation.)

Alternative proof 2: failed

We try to prove (Proposition (commutativity of addition): + =) by induction on *m*.

Induction Base:

(1) $\forall_n (0 + n = n + 0)$.

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: proved

We take in (1) all variables arbitrary but fixed and prove:

(4) $0 + n_1 = n_1 + 0$.

A proof by simplification of (4) works.

Simplification of the lhs term:

$0 + n_1 =$ by (Proposition (20): 20)

n_1⌋

Simplification of the rhs term:

$n_1 + 0 =$ by (Definition (addition): +0:)

n_1⌋

Alternative proof 2: pending

Induction Step:

Induction Hypothesis:

(2) $\forall_n (m_1 + n = n + m_1)$

Induction Conclusion:

(3) $\forall_{n}\ (m_1^{+} + n = n + m_1^{+})$.

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: failed

We take in (3) all variables arbitrary but fixed:

(5) $m_1^{+} + n_2 = n_2 + m_1^{+}$

and simplify it.

Simplification of the lhs term:

$m_1^{+} + n_2 \rfloor$

Simplification of the rhs term:

$n_2 + m_1^{+} =$ by (Definition (addition): + .:)

$(n_2 + m_1)^{+} \rfloor$

Hence, it is sufficient to prove:

(6) $\forall_{n}\ (m_1^{+} + n = (n + m_1)^{+})$.

We try to prove (6) by induction on *n*.

Induction Base:

(7) $m_1^{+} + 0 = (0 + m_1)^{+}$.

A proof by simplification of (7) works.

Simplification of the lhs term:

$m_1^{+} + 0 =$ by (Definition (addition): +0:)

$m_1^{+} \rfloor$

Simplification of the rhs term:

$(0 + m_1)^{+} =$ by (Proposition (20): 20)

$m_1^{+} \rfloor$

Induction Step:

Induction Hypothesis:

(8) $m_1^+ + n_3 = (n_3 + m_1)^+$

Induction Conclusion:

(9) $m_1^+ + n_3^+ = (n_3^+ + m_1)^+$.

We simplify (9).

Simplification of the lhs term:

$m_1^+ + n_3^+ =$ by (Definition (addition): + .:)

$(m_1^+ + n_3)^+ =$ by (8)

$((n_3 + m_1)^+)^+ \rfloor$

Simplification of the rhs term:

$(n_3^+ + m_1)^+ \rfloor$

Hence, it is sufficient to prove:

(10) $((n_3 + m_1)^+)^+ = (n_3^+ + m_1)^+$.

The proof of (10) fails. (No applicable inference rule was found.)

Alternative proof 2: failed

We try to prove (3) by induction on *n*.

Induction Base:

(11) $m_1^+ + 0 = 0 + m_1^+$.

A proof by simplification of (11) works.

Simplification of the lhs term:

$m_1^+ + 0 =$ by (Definition (addition): +0:)

$m_1^+ \rfloor$

Simplification of the rhs term:

$0 + m_1^+ =$ by (Proposition (20): 20)

$m_1^+\rfloor$

Induction Step:

Induction Hypothesis:

(12) $m_1^+ + n_4 = n_4 + m_1^+$

Induction Conclusion:

(13) $m_1^+ + n_4^+ = n_4^+ + m_1^+$.

We simplify (13).

Simplification of the lhs term:

$m_1^+ + n_4^+ =$ by (Definition (addition): + .:)

$(m_1^+ + n_4)^+ =$ by (12)

$(n_4 + m_1^+)^+ =$ by (Definition (addition): + .:)

$((n_4 + m_1)^+)^+\rfloor$

Simplification of the rhs term:

$n_4^+ + m_1^+ =$ by (Definition (addition): + .:)

$(n_4^+ + m_1)^+\rfloor$

Hence, it is sufficient to prove:

(14) $((n_4 + m_1)^+)^+ = (n_4^+ + m_1)^+$.

The proof of (14) fails. (No applicable inference rule was found.)

□

Proof of second conjecture:

Prove:

(Proposition (15): 15) $\underset{n,m}{\forall}\ (n^+ + m = (n + m)^+)$,

under the assumptions:

(Proposition (20): 20) $\underset{m}{\forall}\ (0 + m = m)$,

(Definition (addition): +0:) $\forall_{m} (m + 0 = m)$,

(Definition (addition): + .:) $\forall_{m,n} (m + n^{+} = (m + n)^{+})$.

We prove (Proposition (15): 15) by induction on *n*.

Induction Base:

(1) $\forall_{m} (0^{+} + m = (0 + m)^{+})$.

We take in (1) all variables arbitrary but fixed:

(4) $0^{+} + m_1 = (0 + m_1)^{+}$

and simplify it.

Simplification of the lhs term:

$0^{+} + m_1 \rfloor$

Simplification of the rhs term:

$(0 + m_1)^{+} =$ by (Proposition (20): 20)

$m_1{}^{+} \rfloor$

Hence, it is sufficient to prove:

(5) $\forall_{m} (0^{+} + m = m^{+})$.

We prove (5) by induction on *m*.

Induction Base:

(6) $0^{+} + 0 = 0^{+}$.

A proof by simplification of (6) works.

Simplification of the lhs term:

$0^{+} + 0 =$ by (Definition (addition): +0:)

$0^{+} \rfloor$

Simplification of the rhs term:

$0^{+} \rfloor$

Induction Step:

Induction Hypothesis:

(7) $0^+ + m_2 = m_2{}^+$

Induction Conclusion:

(8) $0^+ + m_2{}^+ = (m_2{}^+)^+$.

A proof by simplification of (8) works.

Simplification of the lhs term:

$0^+ + m_2{}^+ =$ by (Definition (addition): $+$.:)

$(0^+ + m_2)^+ =$ by (7)

$(m_2{}^+)^+$⌋

Simplification of the rhs term:

$(m_2{}^+)^+$⌋

Induction Step:

Induction Hypothesis:

(2) $\underset{m}{\forall}\ (n_1{}^+ + m = (n_1 + m)^+)$

Induction Conclusion:

(3) $\underset{m}{\forall}\ ((n_1{}^+)^+ + m = (n_1{}^+ + m)^+)$.

We take in (3) all variables arbitrary but fixed:

(9) $(n_1{}^+)^+ + m_3 = (n_1{}^+ + m_3)^+$

and simplify it.

Simplification of the lhs term:

$(n_1{}^+)^+ + m_3$⌋

Simplification of the rhs term:

$(n_1{}^+ + m_3)^+ =$ by (2)

$((n_1 + m_3)^+)^+\rfloor$

Hence, it is sufficient to prove:

(10) $\underset{m}{\forall}\ ((n_1{}^+)^+ + m = ((n_1 + m)^+)^+)$.

We prove (10) by induction on *m*.

Induction Base:

(11) $(n_1{}^+)^+ + 0 = ((n_1 + 0)^+)^+$.

A proof by simplification of (11) works.

Simplification of the lhs term:

$(n_1{}^+)^+ + 0 =$ by (Definition (addition): +0:)

$(n_1{}^+)^+\rfloor$

Simplification of the rhs term:

$((n_1 + 0)^+)^+ =$ by (Definition (addition): +0:)

$(n_1{}^+)^+\rfloor$

Induction Step:

Induction Hypothesis:

(12) $(n_1{}^+)^+ + m_4 = ((n_1 + m_4)^+)^+$

Induction Conclusion:

(13) $(n_1{}^+)^+ + m_4{}^+ = ((n_1 + m_4{}^+)^+)^+$.

A proof by simplification of (13) works.

Simplification of the lhs term:

$(n_1{}^+)^+ + m_4{}^+ =$ by (Definition (addition): + .:)

$((n_1{}^+)^+ + m_4)^+ =$ by (12)

$(((n_1 + m_4)^+)^+)^+\rfloor$

Simplification of the rhs term:

$((n_1 + m_4^+)^+)^+ =$ by (Definition (addition): + .:)

$(((n_1 + m_4)^+)^+)^+$ ⌋

□

Proof of theorem:

Prove:

(Proposition (commutativity of addition): + =) $\underset{m,n}{\forall}\ (m + n = n + m)$,

under the assumptions:

(Proposition (15): 15) $\underset{n,m}{\forall}\ (n^+ + m = (n + m)^+)$,

(Proposition (20): 20) $\underset{m}{\forall}\ (0 + m = m)$,

(Definition (addition): +0:) $\underset{m}{\forall}\ (m + 0 = m)$,

(Definition (addition): + .:) $\underset{m,n}{\forall}\ (m + n^+ = (m + n)^+)$.

We prove (Proposition (commutativity of addition): + =) by induction on *m*.

Induction Base:

(1) $\underset{n}{\forall}\ (0 + n = n + 0)$.

We take in (1) all variables arbitrary but fixed and prove:

(4) $0 + n_1 = n_1 + 0$.

A proof by simplification of (4) works.

Simplification of the lhs term:

$0 + n_1 =$ by (Proposition (20): 20)

n_1⌋

Simplification of the rhs term:

$n_1 + 0 =$ by (Definition (addition): +0:)

n_1⌋

Induction Step:

Induction Hypothesis:

(2) $\forall_n (m_1 + n = n + m_1)$

Induction Conclusion:

(3) $\forall_n (m_1{}^+ + n = n + m_1{}^+)$.

We take in (3) all variables arbitrary but fixed and prove:

(5) $m_1{}^+ + n_2 = n_2 + m_1{}^+$.

A proof by simplification of (5) works.

Simplification of the lhs term:

$m_1{}^+ + n_2 =$ by (Proposition (15): 15)

$(m_1 + n_2)^+ =$ by (2)

$(n_2 + m_1)^+ \rfloor$

Simplification of the rhs term:

$n_2 + m_1{}^+ =$ by (Definition (addition): + .:)

$(n_2 + m_1)^+ \rfloor$

□

Example: Proving Booleans of Equalities by Groebner Bases

Formula["Test", any[x, y],

$$((x^2 y - 3x) \neq 0) \vee ((x y + x + y) \neq 0) \vee (((x^2 y + 3x = 0) \vee ((-2x^2 + -7xy + x^2 y + x^3 y + -2y^2 + -2xy^2 + 2x^2 y^2) = 0)) \wedge ((x^2 + -xy + x^2 y + -2y^2 + -2xy^2) = 0))$$

"B1"]

Prove[Formula["Test"], using → {}, by → GroebnerBasesProver]

▪ ProofObject ▪

Note: thousands of (interesting and partly unknown) theorems in (coordinate geometry) can be proved by the Groebner bases (and similar) methods: E.g. Pappus Theorem, Desargues Theorem, ...

The above prover call generates the following proof completely automatically. This proof contains also an explanation of the Groebner bases proof method:

Prove:

(Formula (Test): B1)

$$\forall_{x,y}\ (((x^2 * y - 3 * x) \neq 0) \vee (x * y + x + y \neq 0) \vee ((x^2 * y + 3 * x = 0) \vee ((-2) * x^2 + (-7) * x * y + x^2 * y + x^3 * y + (-2) * y^2 + (-2) * x * y^2 + 2 * x^2 * y^2 = 0)) \wedge (x^2 + (-x) * y + x^2 * y + (-2) * y^2 + (-2) * x * y^2 = 0))$$

with no assumptions.

Proved.

The Theorem is proved by the Groebner Bases method.

The formula in the scope of the universal quantifier is transformed into an equivalent formula that is a conjunction of disjunctions of equalities and negated equalities. The universal quantifier can then be distributed over the individual parts of the conjunction. By this, we obtain:

Independent proof problems:

(Formula (Test): B1.1)

$$\forall_{x,y}\ ((x^2 + (-x * y) + x^2 * y + (-2) * y^2 + (-2) * x * y^2 = 0) \vee ((-3) * x + x^2 * y \neq 0) \vee (x + y + x * y \neq 0))$$

(Formula (Test): B1.2)

$$\forall_{x,y}\ ((3 * x + x^2 * y = 0) \vee ((-2) * x^2 + (-7) * x * y + x^2 * y + x^3 * y + (-2) * y^2 + (-2) * x * y^2 + 2 * x^2 * y^2 = 0) \vee ((-3) * x + x^2 * y \neq 0) \vee (x + y + x * y \neq 0))$$

We now prove the above individual problems separately:

Proof of (Formula (Test): B1.1):

This proof problem has the following structure:

(Formula (Test): B1.1.structure) $\forall_{x,y}\ ((\text{Poly}[1] \neq 0) \vee (\text{Poly}[2] \neq 0) \vee (\text{Poly}[3] = 0))$,

where

$$\begin{aligned}\text{Poly}[1] &= (-3) * x + x^2 * y \\ \text{Poly}[2] &= x + y + x * y \\ \text{Poly}[3] &= x^2 + (-x * y) + x^2 * y + (-2) * y^2 + (-2) * x * y^2\end{aligned}$$

(Formula (Test): B1.1.structure) is equivalent to

(Formula (Test): B1.1.implication) $\underset{x,y}{\forall}\ ((\text{Poly}[1] = 0) \wedge (\text{Poly}[2] = 0) \Rightarrow (\text{Poly}[3] = 0))$.

(Formula (Test): B1.1.implication) is equivalent to

(Formula (Test): B1.1.not-exists) $\underset{x,y}{\nexists}\ (((\text{Poly}[1] = 0) \wedge (\text{Poly}[2] = 0)) \wedge (\text{Poly}[3] \neq 0))$.

By introducing the slack variable(s)

$\{\xi\}$

(Formula (Test): B1.1.not-exists) is transformed into the equivalent formula

(Formula (Test): B1.1.not-exists-slack) $\underset{x,y,\xi}{\nexists}\ (((\text{Poly}[1] = 0) \wedge (\text{Poly}[2] = 0)) \wedge ((-1) + \xi\ \text{Poly}[3] = 0))$.

Hence, we see that the proof problem is transformed into the question on whether or not a system of polynomial equations has a solution or not. This question can be answered by checking whether or not the (reduced) Groebner basis of

$\{\text{Poly}[1], \text{Poly}[2], (-1) + \xi\ \text{Poly}[3]\}$

is exactly $\{1\}$.

Hence, we compute the Groebner basis for the following polynomial list:

$\{(-1) + x^2\ \xi + -x\ y\ \xi + x^2\ y\ \xi + -2\ y^2\ \xi + -2\ x\ y^2\ \xi,\ -3\ x + x^2\ y,\ x + y + x\ y\}$

The Groebner basis:

$\{1\}$

Hence, (Formula (Test): B1.1) is proved.

Proof of (Formula (Test): B1.2):

This proof problem has the following structure:

(Formula (Test): B1.2.structure) $\underset{x,y}{\forall}\ ((\text{Poly}[1] \neq 0) \vee (\text{Poly}[2] \neq 0) \vee (\text{Poly}[3] = 0) \vee (\text{Poly}[4] = 0))$,

where

$$\begin{aligned}\text{Poly}[1] &= (-3)*x + x^2*y\\ \text{Poly}[2] &= x + y + x*y\\ \text{Poly}[3] &= 3*x + x^2*y\\ \text{Poly}[4] &= (-2)*x^2 + (-7)*x*y + x^2*y + x^3*y + (-2)*y^2 + (-2)*x*y^2 + 2*x^2*y^2\end{aligned}$$

(Formula (Test): B1.2.structure) is equivalent to

(Formula (Test): B1.2.implication) $\underset{x,y}{\forall}\ ((\text{Poly}[1] = 0) \wedge (\text{Poly}[2] = 0) \Rightarrow (\text{Poly}[3] = 0) \vee (\text{Poly}[4] = 0))$

.

(Formula (Test): B1.2.implication) is equivalent to

(Formula (Test): B1.2.not-exists)

$\underset{x,y}{\nexists}\ (((\text{Poly}[1] = 0) \wedge (\text{Poly}[2] = 0)) \wedge ((\text{Poly}[3] \neq 0) \wedge (\text{Poly}[4] \neq 0)))$.

By introducing the slack variable(s)

$\{\xi 1, \xi 2\}$

(Formula (Test): B1.2.not-exists) is transformed into the equivalent formula

(Formula (Test): B1.2.not-exists-slack)

$\underset{x,y,\xi 1,\xi 2}{\nexists}\ (((\text{Poly}[1] = 0) \wedge (\text{Poly}[2] = 0)) \wedge \{(-1) + \xi 1\ \text{Poly}[3] = 0, (-1) + \xi 2\ \text{Poly}[4] = 0\})$.

Hence, we see that the proof problem is transformed into the question on whether or not a system of polynomial equations has a solution or not. This question can be answered by checking whether or not the (reduced) Groebner basis of

$\{\text{Poly}[1], \text{Poly}[2], (-1) + \xi 1\ \text{Poly}[3], (-1) + \xi 2\ \text{Poly}[4]\}$

is exactly $\{1\}$.

Hence, we compute the Groebner basis for the following polynomial list:

$$\{(-1) + 3\,x\,\xi 1 + x^2\,y\,\xi 1,\ (-1) + -2\,x^2\,\xi 2 + -7\,x\,y\,\xi 2 + x^2\,y\,\xi 2 + x^3\,y\,\xi 2 + -2\,y^2\,\xi 2 + -2\,x\,y^2\,\xi 2 + 2\,x^2\,y^2\,\xi 2,\ -3\,x + x^2\,y,\ x + y + x\,y\}$$

The Groebner basis:

$\{1\}$

Hence, (Formula (Test): B1.2) is proved.

Since all of the individual subtheorems are proved, the original formula is proved.

□

■ Example: Inventing and Proving Σ-Terms

Paule-Schorn automated conjecture generator / prover:

Formula["SIAM series",

$$\sum_{k=1,\ldots,n} \frac{(-1)^{k+1}\,(4\,k+1)\,(2\,k)!}{2^k\,(2\,k-1)\,(k+1)!\,2^k\,k!}\,]$$

Simplify[Formula["SIAM series"], by → PauleSchorn–Telescope, built–in → Built–in["PauleSchorn"]]

If `−1 + n' is a natural number, then:

$$1+\frac{-(-1)^n\,2^{-2\,n}\,(2\,n)!}{n!\,(1+n)!}$$

```
Prove[Formula["SIAM series"], by → PauleSchorn–Telescope, built–in → Built–in["PauleSchorn"],
  transformBy → Identity, TransformerOptions → {}]
```

Theorem:
If $(-1) + n$ is a natural number, then:

$$\sum_{k=1,\ldots,n} \left(\frac{-(-1)^k\, 2^{-2k}\, (2k)!\, (1+4k)}{k!\, (1+k)!\, ((-1)+2k)} \right) = 1 + \frac{-(-1)^n\, 2^{-2n}\, (2n)!}{n!\, (1+n)!}$$

Proof:
Let Δ_k denote the forward difference operator in k then
the Theorem follows from summing the equation

$$\frac{-(-1)^k\, 2^{-2k}\, (2k)!\, (1+4k)}{k!\, (1+k)!\, ((-1)+2k)} = \Delta_k\left[\frac{(-1)^k\, 2^{1+-2k}\, (2k)!\, (1+k)}{k!\, (1+k)!\, ((-1)+2k)} \right]$$

over the range $k = 1, \ldots, n$.

The equation is routinely verifiable
by dividing the right–hand side by the left–hand side
and simplifying the resulting rational function

$$\frac{\frac{(-1)^{1+k}\, 2^{1+-2(1+k)}\, (2(1+k))!\, (2+k)}{(1+k)!\, (2+k)!\, ((-1)+2(1+k))} - \frac{(-1)^k\, 2^{1+-2k}\, (2k)!\, (1+k)}{k!\, (1+k)!\, ((-1)+2k)}}{\frac{-(-1)^k\, 2^{-2k}\, (2k)!\, (1+4k)}{k!\, (1+k)!\, ((-1)+2k)}}$$

to 1.

```
System`FullSimplify[ (((-1)^(1+k) 2^(1-2 (1+k)) (2 (1+k))! (2+k))/((1+k)! (2+k)! (-1+2 (1+k))) -
    ((-1)^k 2^(1-2 k) (2 k)! (1+k))/(k! (1+k)! (-1+2 k)))/((-(-1)^k 2^(-2 k) (2 k)! (1+4 k))/(k! (1+k)! (-1+2 k))) ]
```

```
1
```

See also talk by Carsten Schneider (PhD thesis, advisor: Peter Paule): implements and significantly extends Karr's method.

■ Example: Calling External Provers (e.g. Resolution)

Links from *Theorema* to various existing theorem provers, for example to the Otter prover:

```
<< "C:\PROGRAM FILES\ATP SYSTEMS\TMA2EXTERNAL\THEOREMA2EXTERNAL.M"
```

Definition["Inclusion",

$$\forall_{A,B}\left((A \subseteq B) \Longleftrightarrow \forall_{x}((x \in A) \Longrightarrow (x \in B))\right) \quad \text{"}\subseteq\text{:"}\Big]$$

Definition["Union",

$$\forall_{A,x}\left((x \in \bigcup A) \Longleftrightarrow \exists_{Y}((x \in Y) \wedge (Y \in A))\right) \quad \text{"}\bigcup\text{:"}\Big]$$

Definition["Powerset",

$$\forall_{A,x}((x \in \mathcal{P}[A]) \Longleftrightarrow (x \subseteq A)) \quad \text{"P:"}\Big]$$

Proposition["Union of powerset includes...",

$$\forall_{A}(A \subseteq \bigcup \mathcal{P}[A]) \quad \text{"}\subseteq P\bigcup\text{"}\Big]$$

ProveExternal[Proposition["Union of powerset includes..."],
using → {**Definition**["Inclusion"], Definition["Union"], Definition["Powerset"]}, by → Otter]

This instruction calls the Otter prover as an external prover and displays the lines of a resolution proof in an extra window.

■ Example: Computation Using Functors

■ A Sparse Polynomials Functor

This definition describes the functor that generates from an arbitrary domain of coefficients C and an arbitrary domain of "power products" (multiplicative terms) T the domain of polynomials over C and T:

Definition["pol", any[C, T],

pol[C, T] = Functor[P, any[c, d, i, $\overline{m}$, $\overline{n}$, p, q, s, t],

$\$ = \langle 0 : P,\ 1 : P,\ + : P \times P \to P,\ - : P \to P,\ - : P \times P \to P,\ * : P \times P \to P \rangle$

$0_P = \langle\rangle$

$1_P = \langle\langle 1_C, 1_T\rangle\rangle$

$\langle\rangle +_P q = q$

$p +_P \langle\rangle = p$

$\langle\langle c, s\rangle, \overline{m}\rangle +_P \langle\langle d, t\rangle, \overline{n}\rangle =$

$\Big\|\ \langle c, s\rangle \frown \left(\langle\overline{m}\rangle +_P \langle\langle d, t\rangle, \overline{n}\rangle\right) \Leftarrow s \geq_T t,$

$\langle d, t\rangle \frown \left(\langle\langle c, s\rangle, \overline{m}\rangle +_P \langle\overline{n}\rangle\right) \Leftarrow t \geq_T s,$

$\left(\left\langle c +_C d, s\right\rangle\right) \frown \left(\langle\overline{m}\rangle +_P \langle\overline{n}\rangle\right) \Leftarrow (c \neq -_C d),$

$\left(\langle\overline{m}\rangle +_P \langle\overline{n}\rangle\right) \Leftarrow \text{otherwise}\Big\|$

$-_P\langle\rangle = \langle\rangle$

$-_P\langle\langle c, s\rangle, \overline{m}\rangle = \langle -_C c, s\rangle \frown \left(-_P\langle\overline{m}\rangle\right)$

$p -_P q = p +_P \left(-_P q\right)$

$\langle\rangle *_P q = \langle\rangle$

$p *_P \langle\rangle = \langle\rangle$

$\langle\langle c, s\rangle, \overline{m}\rangle *_P \langle\langle d, t\rangle, \overline{n}\rangle = \left(\left\langle\left\langle c *_C d, s *_T t\right\rangle\right\rangle +_P \langle\langle c, s\rangle\rangle *_P \langle\overline{n}\rangle\right) +_P \langle\overline{m}\rangle *_P \langle\langle d, t\rangle, \overline{n}\rangle$

]]

Theorema programs are just a special class of predicate logic formulae!

■ Generate 3-Variate Polys over the Integers

The following sequence of Theorema instructions generates the domain of 3-variate polys over the integers and allows to compute with these polys.

```
Theory["F",
        Definition["integers"]
        Definition["power products"]]
        Definition["pol"]
```

```
Definition["D",
        ℙ = pol[integers[], power–products[3]]]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"],
    Built–in["Numbers"], Built–in["Tuples"], Built–in["Sets"], Theory["F"], Definition["D"]⟩]
```

```
Compute[(⟨⟨5, ⟨2, 3, 1⟩⟩, ⟨3, ⟨1, 6, 3⟩⟩⟩ *_ℙ (⟨⟨5, ⟨2, 3, 1⟩⟩, ⟨3, ⟨1, 6, 3⟩⟩⟩ *_ℙ ⟨⟨5, ⟨2, 3, 1⟩⟩, ⟨3, ⟨1, 6, 3⟩⟩⟩))]
```

```
⟨⟨125, ⟨6, 9, 3⟩⟩, ⟨225, ⟨5, 12, 5⟩⟩, ⟨135, ⟨4, 15, 7⟩⟩, ⟨27, ⟨3, 18, 9⟩⟩⟩
```

In Mathematica:

```
(5 x1^2 x2^3 x3 + 3 x1 x2^6 x3^3)^3
```

$$125\,x1^6\,x2^9\,x3^3 + 225\,x1^5\,x2^{12}\,x3^5 + 135\,x1^4\,x2^{15}\,x3^7 + 27\,x1^3\,x2^{18}\,x3^9$$

■ Generate 5-Variate Polys over the Finite Field Modulo 13

```
Theory["F",
        Definition["finite field"]
        Definition["power products"]]
        Definition["pol"]
```

```
Definition["D",
        ℙ = pol[finite–field[13], power–products[5]]]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"],
    Built–in["Numbers"], Built–in["Tuples"], Built–in["Sets"], Theory["F"], Definition["D"]⟩]
```

Generate 5-Variate Polys over the 2 Variate Rational Functions over the Finite Field Modulo 13

```
Theory["F",
        Definition["rational"]
        Definition["power products"]
        Definition["quotient field"]
        Definition["pol"]]
```

```
Definition["D",
        ℙ = pol[
      quotient–field[
        pol[
          Galois–field[13],
          power–products[2]]],
      power–products[5]]]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"],
    Built–in["Numbers"], Built–in["Tuples"], Built–in["Sets"], Theory["F"], Definition["D"]⟩]
```

Conclusion

I believe that, going into the direction of systems like *Theorema*, the following is / will soon be possible:

- Computer-support of all aspects of doing mathematics will reach higher and higher levels (by the meta-level principle) including inventing, exploring, proving.
- In particular, nonalgorithmic and algorithmic mathematics will be supportable in one common logical and software-technological frame (in other words, mathematics, and computer science will reconcile).
- Mathematical software systems, in addition to providing algorithm libraries, will (have to) provide huge mathematical knowledge libraries. Building up and using such libraries, essentially, is a task of formal logic. Hence, computer-supported derivation of theorems from knowledge bases will be a future key tool.
- For building up and maintaining such mathematical knowledge libraries, an international consortium is necessary ("Bourbaki Project in the Digital Age").

- The education of mathematicians will have to shift towards including in-depth training on the formal / logical aspects of mathematics.